COBB

June 1992
Vol. 3 No. 6

INSIDE DOS™

Tips & techniques for MS-DOS & PC-DOS

How your computer pulls itself up by its bootstraps

VERSIONS 2.1-5.0

Sometimes it seems that computer jargon takes simple words and gives them mysterious meanings. For example, we've all heard programmers complain about "a bug in the program." But did you know that, once upon a time when computers used vacuum tubes instead of chips, real bugs could wreak havoc on a program if they bounced off a vacuum tube? Decades later, we still talk about errors in computer code as if they were pesky little insects.

Another term you might wonder about is "boot" (or "booting up"). This term is actually a shortened form of the old saying "pulled himself up by his own bootstraps"—an expression used to describe the type of person we might call a "self-starter." The analogy is appropriate when you consider that a computer starts itself from nothing as it boots. In fact, your PC is nothing but a beige hunk of metal and silicon until electricity animates it. And, even after you turn on the juice, the only instructions it can read at first are those wired into its microprocessors (better known as "chips").

Unfortunately, it's hard to tell exactly what's going on when you boot up your computer, either by turning it on (a "cold boot") or by pressing [Ctrl][Alt][Delete] (a "warm boot"). You hear a few beeps and messages scroll across the screen at unreadable speeds. To help you better understand what happens, we'll take a look at how your computer boots, and we'll show you some sample messages produced in the process. (Of course, your PC might display different names and numbers from the PC used in our example, but it will probably display the same categories of information.) As you'll see, most computers go through three groups of tasks as they boot:

- checking out the integrated circuits, or chips, installed on its motherboard
- configuring the system as instructed by the CONFIG.SYS file
- executing any DOS commands you've placed in its AUTOEXEC.BAT

By understanding more about what goes on during these three stages, you'll understand more about how the basic parts of your computer work together. Later, if you run into problems when you boot up after adding new devices or new software to your system, you'll have a better idea where and how to begin looking for the cause. Along the way, we'll provide some basic definitions for the components in computers, which should help you when you're shopping for your next PC. Let's get started by looking at the most fundamental part of your computer: the chips.

Powering up the chips

When you first turn on your computer, you send current through its power supply and into a large circuit board called the *motherboard*. The motherboard then distributes the power to integrated circuits, which are better known as chips. The motherboard is in many ways the spine of the computer. It allows information to travel among all the components that you plug into it. If you looked inside your computer, you'd see that the motherboard has several chips plugged into it, as well as a circuit card that links your monitor to the inside of your computer.

Display memory

If you watch closely when you boot up, you'll probably see a few messages describing the chips inside your computer. With some machines, the first message you see displays the manufacturer, type, and amount of display

Continued on page 11

IN THIS ISSUE

A Publication of The Cobb Group

INSIDE DOS™

Inside DOS (ISSN 1049-5320) is published monthly by The Cobb Group.

Prices	Domestic .... $59/yr. ($7.00 each) Outside the US .... $79/yr. ($8.50 each)
Address	The Cobb Group 9420 Bunsen Parkway, Suite 300 Louisville, KY 40220
Phone	Toll free .... (800) 223-8720 Local .... (502) 491-1900 FAX .... (502) 491-8050
Staff	Editor-in-Chief .... Jim Welp Editor .... Suzanne Thornberry Consulting Editors .... Mark W. Crane Tim Landgrave Contributing Editor .... Van Wolverton Editing .... Clyde Zellers Polly Blakemore Cecilia Crosby-Lampkin Production .... Debbie Lane Design .... Karl Feige Publications Manager .... Elayne Noltemeyer Publisher .... Douglas Cobb

Address correspondence and special requests to The Editor, *Inside DOS*, 9420 Bunsen Parkway, Suite 300, Louisville, KY 40220. Address subscriptions, fulfillment questions, and requests for bulk orders to Customer Relations, at the same address.

Postmaster: Send address changes to *Inside DOS*, P.O. Box 35160, Louisville, KY 40232. Second class postage is paid in Louisville, KY.

Conventions

To avoid confusion, we would like to explain a few of the conventions used in *Inside DOS*.

When we instruct you to type something, those characters usually appear on a separate line along with the DOS prompt. The characters you type will appear in red, while the characters DOS displays will appear in black.

Occasionally, we won't display the command you type on a separate line. In these cases, we'll display the characters you type in italics. For example, we might say, "Issue the command *dir *.txt* at the DOS prompt." Although DOS is not case-sensitive, we'll always display the characters you type in lowercase.

When we refer to a general DOS command (not the command you actually type at the DOS prompt), we'll display that command name in all caps. For example, we might say, "You can use either the COPY or XCOPY command to transfer files from one disk to another."

Many commands accept parameters that specify a particular file, disk drive, or other option. When we show you the form of such a command, its parameters will appear in italics. For example, the form of the COPY command is

```
copy file1 file2
```

where *file1* and *file2* represent the names of the source file and the target file, respectively.

The names of keys, such as [Shift], [Ctrl], and [F1], appear in brackets. When two keys must be pressed simultaneously, those key names appear side by side, as in [Ctrl][Break] or [Ctrl]Z.

LETTER

Creating a batch file that runs a second program after exiting the first

VERSIONS 2.1-5.0

I use a bill payment program to keep track of my personal checking account and a separate accounting program to track my assets and liabilities. With this setup, I almost always update the accounting program after the bill payment program. As you might guess, I want to create a batch file that would run the checking program and, when I quit, automatically run the accounting program.

Can you explain how to create a batch file that will run one program and then run another after you quit the first? Such a batch file would be most useful if it would let you choose whether to run the second program or quit after the first one.

Neville Wilson
Brooklyn, New York

Many of us have work routines that require running programs sequentially. You can automate running these programs by creating a "quick and dirty" batch file.

The technique

When you're creating a batch file to run two programs, we suggest you begin with the *@echo off* command. If you're using a version prior to DOS 3.3, leave off the @ symbol from the *echo off* command. This command prevents DOS from echoing each of the batch file commands to the screen. (As with any batch file, you can enter the commands using Edlin, DOS 5.0's Editor, or a word processor that lets you save a file in plain ASCII format.) After you turn off echoing, you enter the commands needed to run the first program. Usually, you'll change to the directory that contains the program and then, on the next line, enter the command that runs the program.

After running the first program, you'll want the batch file to pause so that you can choose to return to DOS or to run the second program. Although the PAUSE command will automatically present the message *Press any key to continue…* (or *"Strike a key when ready..."* in previous versions of DOS), you can add a more detailed message by typing *echo*, followed by your message, on the line before the PAUSE command.

Once you've set up the ECHO and PAUSE commands, you enter the commands that run the second program. (Again, you'll probably use one command to change to the directory containing the program and another to execute the program.) Finally, as a bit of housekeeping, you can add a command to change back to the root directory when you finish the second program.

Creating FRIDAY.BAT

Now that we've outlined the general steps for creating a batch file that runs two programs, let's take a look at creating a sample batch file. (Of course, you'd substitute the directories and program names for the applications you want to run—you don't have to use Excel and Word.) Suppose you create a report every Friday that contains an updated Excel spreadsheet and a Word document. You can create FRIDAY.BAT, shown in Figure A, to run Excel and then run Word after you exit Excel.

Figure A

```
@echo off
cd \excel
excel
echo Press [Ctrl]C, then Y to return to DOS. Or, to run Word,
pause
cd \word
word
cd \
```

FRIDAY.BAT will automatically run Word after you quit Excel.

You begin the batch file with the command to turn off echoing:

```
@echo off
```

(As we said earlier, leave off the @ symbol if you're using a version of DOS prior to 3.3.) Then, on the next lines, simply type the commands to run Excel:

```
cd \excel
excel
```

The next section of the batch file contains a brief message explaining how to return to DOS, along with the PAUSE command. As you may know, whenever you use a PAUSE command, DOS will PAUSE the system and present the *Press any key to continue...* message. If you want to include an additional message, you can insert it on the line *before* the PAUSE command. To tell DOS to display your message, type *echo*, followed by the text you want to appear. When you run the batch file, DOS will display your message before it displays the automatic *Press any key to continue...* message.

Now, let's look at how we can use the ECHO and PAUSE commands in FRIDAY.BAT. On the line after the *excel* command, you need to add the ECHO command followed by the message *Press [Ctrl] C, then Y to return to DOS. Or, to run Word,*. In FRIDAY.BAT, you can echo a message to remind users that they can exit to DOS by pressing [Ctrl]C (or [Ctrl][Break]). This message also should remind users to press Y to answer yes to the prompt *Terminate batch job (y/n)*, which DOS presents any time you press the break keys. You also can tell users that they'll run Word if they don't terminate the batch file. After typing your message, place a PAUSE command on the next line. The statement below will remind you—or other users—how to exit to DOS, as well as what program you'll run if you continue:

```
echo Press [Ctrl]C, then Y to return to DOS. Or, to
    run Word,
pause
```

After displaying your message—*Press [Ctrl]C, then Y to return to DOS. Or, to run Word,*—DOS will display on the next line *Press any key to continue....*

Once you've typed your message and the PAUSE command, you type the commands needed to run Word:

```
cd \word
word
```

Since you probably don't want to be in the C:\WORD directory when you exit, you can add the command

```
cd \
```

to the end of FRIDAY.BAT to return you to the root directory.

Once you've finished typing all of the commands for FRIDAY.BAT, save the file in a directory on your path. If you're not sure which directories are on your path, you can simply type *path* at the DOS prompt, then press [Enter]. DOS will display the directories in the current path. As a rule, we suggest creating a C:\BATCH directory, placing it in your path, and storing all batch files there (except AUTOEXEC.BAT, which goes in the root directory).

Running FRIDAY.BAT

Running FRIDAY.BAT is simple. Just type *friday*, then press [Enter]. DOS will run Excel. After you've finished working with your spreadsheet, quit Excel. DOS will present the message

```
Press [Ctrl]C, then Y to return to DOS. Or, to run
    Word,
Press any key to continue...
```

When you press any character key (keys such as [Ctrl] and [Shift] don't count), you'll run Word. On the other hand, when you press [Ctrl]C, DOS will present the prompt

```
Terminate batch job (y/n)
```

When this happens, you can press Y to return to the DOS prompt without running Word.

If hardware's hard, shouldn't software be easy?

The steep downward spiral in hardware prices creates bargains galore, but deepens the hesitant buyer's dilemma: Should I buy now, or wait a few months and either pay less or get a lot more machine for my money? There's no easy answer to these questions, of course, but a quick one is "buy now if you need a machine now, wait if you don't need a machine now." (The answer's quick because it isn't very helpful. Sure, prices will probably continue to fall, but if you wait to buy, you give up the advantage of using that bigger, faster machine now.)

Software is usually a bit easier to choose. The Big Three programs—word processing, spreadsheet, and database—still account for the lion's share of computer use; toss in desktop publishing/graphics, and you're talking at least 80 percent. There haven't been any new, gotta-have-it-now application programs for years.

DOS itself poked along from version 1.0 to version 4.01 without arousing anything resembling excitement from either users, sellers, or computer journalists. A new release came along every 18 months or so. DOS worked well, and it kept pace with mainstream hardware developments because IBM defined both the hardware and software standards for the first seven or eight years after the PC was introduced in 1981.

Changes, changes, changes

Starting a couple of years ago, operating systems became the hot topic among computer aficionados. Windows 3.0 and DOS 5.0 were the cover stories in one magazine after another; columnists periodically wondered if OS/2 would ever become real, and industry analysts pondered the fate of the business relationship between Microsoft and IBM. Interest waned for a while after Microsoft released Windows 3.0 in 1990 and DOS 5.0 in 1991, but the spotlight is on operating systems again. This time, both the questions and the cast of characters have changed a bit.

Windows 3.0, the product that finally delivered what Windows promised, has now been replaced by Windows 3.1, which doesn't add a lot but certainly tidies things up nicely. IBM touts OS/2 2.0 as a better DOS than DOS, a better Windows than Windows. In the meantime, there's DR DOS 6.0 from Digital Research by way of Novell, a DOS clone to go along with your PC clone. Not to mention faint rumors from Microsoft hinting of MS-DOS 6.0.

What's going on here? All of these operating systems will run your DOS programs, such as WordPerfect or Lotus 1-2-3, and OS/2 will run your Windows programs, too. Which operating system (or, as Windows is called, *operating environment*) should you be using? Does it matter? Should you even care?

A quick, fond look back

This sort of pondering developed slowly. When DOS was introduced in 1981, it was, for all practical purposes, the only game in town; it didn't cost much and didn't make many demands on your system. It didn't do much, either, of course, so each release offered new features that required more memory, more disk space, and—in some instances—ran more slowly. That's progress.

Version 1.0 begat version 1.1, which begat 2.0, which begat 2.1, and so forth until version 5.0 came forth in the spring of 1991. With the exception of version 4.0, most users upgraded at each major release (marked by an increase in the number to the left of the decimal point). Our favorite operating system grew from two diskettes (a whopping 160 Kb each) in version 1.0 to as many as seven diskettes of 360 Kb each. There was no such thing as a hard disk when version 1.0 came out: DOS ran from those selfsame two 160 Kb diskettes. Now DOS takes up more than two megabytes on your hard disk. You can't just copy the files to your hard disk; you've got to install DOS with a special program. More progress.

Windows came along in 1985, a DOS add-on that purported to make a PC look (if not act) like a Macintosh. That first version of Windows gobbled memory and disk space, slowed down performance, broke down at the slightest provocation, and wasn't particularly easy to use. Compounding the problems, few programs were available that took advantage of its graphic capabilities. But like DOS, Windows evolved, and now it's faster, easier to use, doesn't break down nearly as often, boasts a host of dazzling applications, and lets you switch back and forth (almost) effortlessly among your DOS applications.

And now there's OS/2 2.0, an entirely different operating system that IBM hopes will lure us all away from both DOS and Windows. It, too, offers a graphical interface and all but requires a mouse. However, OS/2 2.0 is even more demanding in the hardware department: It won't even run unless your system has an 80386 or 80486 processor, and it performs sluggishly on anything less than a 25 MHz 80386 with six megabytes of memory and an 80-megabyte hard disk.

So, you can choose from four operating environments for your PC:

- MS-DOS
- DR DOS
- DOS+Windows
- OS/2

MS-DOS and DR DOS are virtually identical; DR DOS, which is up to version 6.0, offers a few more features than MS-DOS 5.0. Windows doesn't replace DOS; it runs on top of it. OS/2 replaces DOS entirely—it's an alternative to the DOS+Windows combination.

How do they compare?

Probably the most important characteristic of these operating-system alternatives to most users is the size of machine they require to run effectively. DOS runs comfortably on a minimum machine, although to take advantage of the memory management capabilities of version 5.0, you should avoid the 80286. A good system for running DOS applications is a 16 MHz 80386SX; toss in a couple of megabytes of memory, a 40-megabyte hard disk, and a super VGA display, and you're looking at $1,200 or so.

Windows would run acceptably fast on that $1,200 80386SX system, but you'd be a lot happier with a 25 MHz 80386 with four megabytes of memory and an 80-megabyte hard disk. This system runs about $1,900.

OS/2 doesn't necessarily need a faster processor than Windows, but you'll have to add memory and hard disk capacity to keep it humming smoothly. A reasonable package would be a 33 MHz 80386 machine with eight megabytes of memory and a 120-megabyte hard disk. This system will set you back around $2,300. Make it an 80486—it *is* faster—and you'll spend about $3,000.

(These prices, by the way, represent an average mail-order system price. If you buy from a computer store, figure on paying at least 25 percent more.)

Who uses them now?

Depending on whose numbers you believe, DOS runs on 50 to 75 million computers worldwide, in every possible environment from elementary school classrooms to international corporations. DR DOS is probably used on a couple of million machines.

Windows is much less widely used. Microsoft says it has shipped millions of copies, but it's unclear how many of those copies are actually being used as the computer's full-time operating environment. Many of those copies are a limited version (called *run-time*) that can only be used to run the application with which they were shipped; from the user's standpoint, Windows is part of the application. Many PC-compatible computers come with Windows already installed, but if the buyer doesn't want it (or doesn't want to take the time to learn it), Windows gets erased. Some users bought Windows 3.0 or 3.1 because of all the publicity, but use it to run only one application or don't even run it at all. It's possible that Windows runs as a full-time operating environment on no more than a million machines.

Even fewer systems run OS/2. Perhaps a quarter of a million people use it, mostly in large corporations, government agencies, educational institutions, and software development companies. Many of these people use OS/2 primarily because their systems are wired up as a local area network (LAN), which, in turn, is linked to a mainframe computer. The advantage offered by the memory-management and multitasking features of OS/2 in a LAN outweigh the system cost.

Where are we going?

Version 5 was the most eagerly anticipated—and most successful—version of DOS since version 2. The next release will probably incorporate more of the features of Windows—particularly improvements in memory management and task-switching capabilities. There's a chance that DOS itself may become a more graphical system, with the Shell (or its evolutionary successor) becoming the primary means of controlling DOS and the command line becoming the rare exception. DR DOS may well develop along similar lines.

Windows 3.1 further increases the appeal established by Windows 3.0. Improvements in memory management and overall system performance have dramatically reduced the disadvantage created by previous versions of Windows. Thousands of Windows applications are available, and new ones pop up every day. Microsoft vows to continue improving Windows and insists that future versions will have all the capabilities of OS/2 and more.

OS/2 is still evolving. IBM has expended enormous effort to make it a viable alternative to DOS and Windows, but it requires an expensive system and it is difficult to install. Only a few OS/2 applications are available, but IBM is doing everything it can to encourage software developers to convert applications to OS/2 or to write new ones. OS/2 still seems to be the province of large installations with deep pockets and resident experts who can master its foibles, but IBM seems committed to the strategy of making OS/2 all things to all users at the expense of DOS and Windows.

What does it all mean?

If you work in a very large company, government agency, or educational institution—especially if your machine is part of a local area network linked to a mainframe—there's a good chance that you'll be using OS/2 by the end of 1993; if not, you'll probably be using Windows. You most likely won't make the decision yourself.

If you work in a small- to medium-sized company, agency, or school, you'll probably be using Windows by the end of 1993. If your machine is part of a LAN—especially one that's linked to a mainframe—you might be using OS/2. Again, you probably won't make the decision yourself.

But if you own or work in a very small business (one to three computers) or use your computer at home, the decision most likely will be yours to make. Chances are you won't need OS/2, even if your system is powerful enough to run it effectively. Whether you should add Windows depends on the applications you plan to use, the size of your system, and how much time you're willing to spend learning to use a new software system.

If you use only a few character-based applications, such as Lotus 1-2-3 Release 2.2 or Microsoft Word, you are probably better off sticking with unadorned DOS—especially if you have anything less than a 16 MHz 80386SX machine with two megabytes of memory, a 40-megabyte hard disk, and a super VGA monitor. However, if you want to use Windows-specific applications like Excel or PageMaker, you must run Windows on top of DOS.

If you use several DOS programs and are considering installing Windows simply to get multitasking, don't overlook the simpler task switcher in the DOS 5 Shell. It's just as quick or quicker, and it offers you everything you need unless you really need multitasking—the ability to have one or more programs continue running in the background while you do something else.

If you're interested in new programs, you'll have to face the fact that more and more of the exciting new applications will be written for Windows or OS/2. Keep in mind, however, that a system that gives acceptable performance when running Windows costs $750 to $1,000 more than a system that runs DOS well. If you want to run OS/2, add another $500 to $1,000.

DOS is here to stay, for now

If you plan to continue using plain old DOS—or if you don't want to spend the money for a system with enough horsepower to run Windows or OS/2—don't worry, you have plenty of company. Microsoft and IBM are eager to sell lots of copies of Windows and OS/2, but they aren't about to abandon tens of millions of DOS users. You can look for additional versions of DOS in the future that will continue to improve *its* memory management and other features—and probably make it look more like a mini-Windows. However, you can rest assured that the mainstream applications that run so well under DOS today will be supported for years to come. ■

DOS 5 TIP

VERSION 5.0

Reinstalling selected files

By Mark Kimbell

Have you ever issued a DOS command only to be admonished by the

```
Bad command or file name
```

error message? After you double-check your directory, command syntax, and other likely causes of such a message, you might discover that you've deleted that DOS 5 command from your hard drive. Perhaps you deliberately deleted the command to save disk space or to prevent someone from making a catastrophic mistake (like reformatting your hard drive). Or you might have been careless with your wildcard characters and inadvertently deleted the command while sprucing up your hard disk. Whatever the reason, you'll now probably want to replace the missing file with another copy from your original DOS 5 disks.

Compressed files

Unfortunately, the process of restoring deleted DOS files isn't quite as easy with DOS 5 as it was with previous versions. Since DOS 5 is a fairly large program, Microsoft compresses DOS 5 files so they can pack the program onto fewer disks. Like a collapsed umbrella, a compressed file is great for minimizing storage space, but you must expand it before you can use it. That's why DOS 5's installation program expands all the compressed files automatically when you install DOS 5. At that point, the files that began as compressed files on your DOS 5 disks become fully functional files on your hard drive.

Obviously, you don't want to reinstall DOS 5 just to make an executable copy of a single compressed file. For that reason, DOS 5 offers you a more convenient and selective alternative.

Using the EXPAND command

DOS 5's EXPAND command allows you to expand the DOS 5 compressed file of your choice. That way, you can make an executable copy of a compressed file without reinstalling the whole program. You begin by inserting into your floppy drive the DOS 5 disk containing the compressed file you want to expand. Then, you issue the EXPAND command, specify the name of the compressed file, and then specify the path and name of the expanded copy.

To help you determine which DOS 5 disk contains the compressed file you want to expand, DOS 5 includes a road map in a file named PACKING.LST. To view its contents, issue the command

```
C:\DOS>type packing.lst | more
```

When you do this, DOS 5 will scroll through a list of 117 files, 104 of which are compressed. DOS 5 identifies a compressed file by replacing the last character of its file extension with the underscore character (_). For instance, the compressed version of FORMAT.COM appears in PACKING.LST as FORMAT.CO_. Of course, when you expand the compressed file, you replace the underscore character with the appropriate letter in the file extension. Once you've identified the name and location of the file you want to expand, the rest is easy. For example, to copy to your C:\DOS directory a working version of FORMAT.COM, insert Disk 1 of your DOS 5 Upgrade package into your floppy drive (we'll use B:) and issue the command

```
C:\>expand b:format.co_ c:\dos\format.com
```

In a few seconds, DOS will place an expanded copy of FORMAT.COM in your C:\DOS directory.

Notes

When you use DOS 5's EXPAND command, you'll want to keep a few things in mind. First, you can assign any name you want to the expanded file. For instance, to use the name CAREFUL instead of FORMAT in the previous example, you'd simply issue the command

```
C:\>expand b:format.co_ c:\dos\careful.com
```

Second, although you can assign any name you want to the expanded file, you should use the correct file extension. Table A lists the compressed file extensions, along with the corresponding expanded file extensions.

Table A

Compressed extension	Expanded extension
ba_	bas
co_	com
cp_	cpi
ex_	exe
gr_	grb
hl_	hlp
in_	ini
sy_	sys
vi_	vid

When you expand a compressed file, you should replace the underscore character in the file extension with the correct letter.

Third, DOS 5 lets you omit the name of the expanded file from the EXPAND command if you want, as long as you include a path. For instance, you can issue the command

```
C:\>expand b:format.co_ c:\dos
```

to expand a copy of FORMAT.CO_ and place it in your C:\DOS directory. However, if you do this, DOS 5 will assign the name FORMAT.CO_ to the expanded file, which will prevent DOS 5 from recognizing it as a COM file. As a result, you'll have to use the RENAME command to replace the CO_ file extension with COM before you can use the FORMAT command.

Finally, EXPAND.EXE isn't compressed. As a result, if you need a backup copy of this file, you can simply copy it from your DOS 5 disk.

Mark Kimbell is the editor-in-chief of several journals from The Cobb Group, including 1-2-3 User's Journal.

Running AUTOEXEC.BAT instead of rebooting

Have you ever issued a new PATH command when you're experimenting with DOS, then wanted your old path back? You might try to type *path*, followed by the directories you usually use as your system's default path. Unfortunately, it's easy to make a mistake when you're typing all those pathnames, colons, and semicolons. In desperation, you might decide to reboot to restore the path you set in your AUTOEXEC.BAT file. However, you won't be able to use your PC as it reboots.

A quicker way to restore any of the commands you issued in the AUTOEXEC.BAT file is to run AUTOEXEC again. As you'd expect, you do this by typing the command

```
C:\>autoexec
```

and then pressing [Enter]. Your computer will run all of the commands in the AUTOEXEC.BAT file in an instant, and you can continue working with your restored settings.

This technique isn't limited to resetting your path, although that's a fairly common application for it. You also can run AUTOEXEC to reset your system prompt or to re-execute any other command the file contains.

Adding a header line to your DOS screen

In the March and May 1992 issues of *Inside DOS*, we presented some techniques for jazzing up your DOS screen by using ANSI codes along with the PROMPT command. In the March article "Fun With the DOS Prompt," we showed you how to use metastrings to make your prompt more informative. In May, we showed you how to change screen colors in "Using the PROMPT Command to Color Your DOS Screen."

In this article, we'll present a variation on both of those themes. We'll create HEADER.BAT, a batch file that creates for your screen a header line displaying the current date, time, and any text string you choose. Not only does HEADER.BAT build a more useful prompt, but creating it affords you the opportunity to issue what's probably the longest DOS command you will ever type.

Introducing HEADER.BAT

You've probably seen data entry screens that have a title or header line on the top row. Typically, this header line displays the company or module name along with the current date and time.

If you've loaded ANSI.SYS from your CONFIG.SYS file, you can add a similar header line to your DOS screens by running HEADER.BAT, shown in Figure A. (For step-by-step instructions on loading ANSI.SYS from your CONFIG.SYS file, see last month's article "Using the PROMPT Command to Color Your DOS Screen.")

After you've created HEADER.BAT, you can display the new prompt by typing

```
C:\>header
```

When you do, your screen will look like the one in Figure B. To customize HEADER.BAT for your company, edit the first line of the batch file and replace the text *The Cobb Group* with the name of your company.

Let's take a closer look at HEADER.BAT. The first line

```
SET NAME=The Cobb Group
```

defines the environment variable that holds your company's name (or any other text string you choose). The second line

```
prompt $e[s$e[H$e[1B$e[K$e[H$e[K$t$h$h$h$e[1;33H
   $e[1;37;41m%name%$e[0;37;40m$e[1;67H$d$e[u$p$g
```

redefines the DOS prompt; the final line clears the screen. The PROMPT command—chock-full of ANSI codes—is a bit intimidating at first glance. To understand how this prompt works, you must examine it one command at a time. Table A lists the commands from the prompt definition in HEADER.BAT. Let's take a look at a couple of these ANSI commands in more detail so you can better customize the batch file.

Customizing HEADER.BAT

You'll want to substitute another text string for the header text *The Cobb Group*. As we mentioned, you can easily change the text by editing the first line of the batch file. However, depending on the length of your text string, you may find it's not centered. That's because the command *$e[1;33H* moves the cursor to the 33rd position on the top row of your screen, the appropriate position for centering the text *The Cobb Group*. Therefore, if your text string is shorter or longer, the header text won't be in the center of the top row. You'll need to modify this command depending on the length of your text string. For instance, if you changed the first line of the batch file to

```
SET NAME=The Company With A Really Long Name
```

you'd want to change the command *$e[1;33H* to *$e[1;22H* so DOS would center the longer text string.

Figure A

```
SET NAME=The Cobb Group
prompt $e[s$e[H$e[1B$e[K$e[H$e[K$t$h$h$h$e[1;33H$e[1;37;41m%name%$e[0;37;40m$e[1;67H$d$e[u$p$g
cls
```

HEADER.BAT adds a header line to your DOS screen.

Figure B

```
13:23:51                        The Cobb Group                        Mon 06-01-1992

C:\>_
```

Here's how your screen looks after running HEADER.BAT.

Table A

Command	Description
`$e[s`	save current cursor position
`$e[H`	move to upper-left corner of screen
`$e[1B`	move down one row
`$e[K`	clear entire row
`$e[H`	move to upper-left corner of screen
`$e[K`	clear entire row
`$t`	display current time
`$h$h$h`	backspace three times to erase hundredths of seconds from time string
`$e[1;33H`	move to 33rd position of top row
`$e[1;37;41m`	change colors to white on red
`%name%`	display environment string *name*
`$e[0;37;40m`	change colors to white on black
`$e[1;67H`	move to 67th position of top row
`$d`	display current date
`$e[u`	restore cursor position saved with *$e[s*
`$p$g`	display drive and directory prompt

Also, notice that the *$e[1;37;41m* command changes the color of the header text to white letters on a red background. You might want to change the screen to some other color. For a complete discussion of changing screen colors with the PROMPT command, including a table listing the ANSI codes you'll need, see last month's article, "Using the PROMPT Command to Color Your DOS Screen."

As always, there are a few pitfalls to avoid when customizing this batch file. First, if the text string is too long, the PROMPT command might go beyond 127 characters—the maximum command-line length. Second, if ANSI.SYS isn't loaded in your CONFIG.SYS file, the prompt won't work and the *$e[* commands will be visible in the resulting prompt. Finally, there may not be enough room in the environment to hold the entire prompt. When this happens, the prompt may be truncated. Since the default environment size is only 256 bytes, you may need to add the command

```
SHELL=C:\COMMAND COM /p/e:1024
```

to your CONFIG.SYS file. This command expands the environment so it's large enough to hold the long prompt along with PATH, COMSPEC, TEMP, and all the other environment variables. The SHELL command locates the command interpreter, COMMAND.COM, and the COMMAND command's */e:* switch sets the environment size in bytes. Remember that you must reboot before the change to CONFIG.SYS will take effect. ■

TROUBLESHOOTING TIP

VERSIONS 2.1-5.0

Reading the messages generated by your CONFIG.SYS

Do you like to tinker with your system? Perhaps you've installed a new peripheral, such as a CD-ROM drive. Or, maybe you've installed software that requires you to use random access memory (RAM) in a certain way. If you've tried these do-it-yourself projects, you've probably had to modify your CONFIG.SYS file. If you make the changes correctly, you should be able to reboot and use the new software or device. But sometimes your changes don't work at first. At times like this, saving the messages generated by your CONFIG.SYS can help you quickly find out which statement is causing the problem.

Recently, subscriber Michael DiPiazza was troubleshooting his system and called to ask if we knew a way to print the messages generated as DOS executes the instructions in the CONFIG.SYS file. As Mr. DiPiazza discovered, it's difficult to capture this information. For example, you might try pressing [Shift][PrintScreen] to send the information onscreen to your printer. Unfortunately, nothing will happen when you press [Shift][PrintScreen] until *after* DOS processes the instructions in your CONFIG.SYS file. Prior to that, DOS hasn't loaded COMMAND.COM, which allows it to interpret the [Shift][PrintScreen] keystroke.

Pausing scrolling

Let's take a look at a keystroke you can use earlier in the boot-up process. As you may know, you can press [Ctrl]S to pause the display's scrolling temporarily. Pressing [Ctrl]S a second time allows your computer to continue with whatever task it was doing. This pause technique offers a bit more potential than [Shift][PrintScreen], since you can use the command just after your system's RAM check. (We explain this process in more detail in "How Your Computer Pulls Itself Up by Its Bootstraps" on page 1.)

In fact, if you press [Ctrl]S just after the computer finishes its RAM check and after you hear the computer check its drives, you'll be able to read the messages generated by your computer's chip checks. When you're ready to continue booting up, simply press [Ctrl]S again. Using this technique, we displayed the following information about the chips in a Zeos 386 computer:

```
SpeedSTAR VGA Vers. 3.00 (c) Diamond Computer
Systems, Inc
512KB display memory installed

Phoenix 80386 ROM BIOS PLUS Version 1.10.10
Copyright (C) 1985-1989 Phoenix Technologies Ltd.
All Rights Reserved

MYLEX Corporation
386

640K Base Memory, 03456K Extended
```

After these messages describing the chips in your PC, you'll probably see a fragment of a message generated by your CONFIG.SYS file. For example, if the first command in your CONFIG.SYS installs the HIMEM.SYS driver, you might see part of a message that begins with

```
HIMEM: DOS XMS Driver, Version 2.60 -
```

Of course, if you press [Ctrl]S after DOS executes the remaining commands in your CONFIG.SYS file, you'll be able to read the rest of the messages. But this is a haphazard way of capturing the information. Even if your reflexes are those of an Indy driver, a slight delay between when you press the keys and when DOS responds makes it difficult to freeze the display at the perfect instant.

A more reliable method

While we don't have a magical command that will tell your CONFIG.SYS file to generate a listing of its messages, we can suggest a reliable technique. Place a PAUSE command in the first line of your AUTOEXEC.BAT file. You can make the changes with Edlin, DOS 5.0's Editor, or any word processor that allows you to save files in ASCII format. Once you've opened your AUTOEXEC.BAT file, simply move your cursor to the beginning of the file, type *pause*, then press [Enter] to create the new line. After making your change, save the new version of the AUTOEXEC.BAT file and return to DOS.

Now, to see the messages generated by your CONFIG.SYS file, press [Ctrl][Alt][Delete] to reboot. After carrying out the commands in your CONFIG.SYS file, your system will pause and display the message

```
Press any key to continue...
```

(Or, if you're using an earlier version of DOS, you'll see the message *Strike a key when ready....*) At this point, you can read the messages generated by your CONFIG.SYS file. If you'd like, you can print a copy of the messages by making sure your printer is online, then pressing [Shift][PrintScreen]. (With most printers, you'll have to press the form feed button to eject the page.) When you're finished reading or printing the messages, press any key to continue booting your system.

Now, let's take a look at a sample CONFIG.SYS file and see what messages it generates. Suppose you entered into your CONFIG.SYS file the following commands:

```
files=20
buffers=30
device=c:\dos\himem.sys
dos=high
device=c:\dos\asni.sys
```

Because the ANSI.SYS driver you're trying to load in line 5 is misspelled as *ASNI.SYS*, DOS won't be able to install the ANSI.SYS driver. By placing the PAUSE command in the AUTOEXEC.BAT file, you'll halt the scrolling so that you can find the error. If you rebooted with this configuration, you'd see that the CONFIG.SYS file generated the following messages:

```
HIMEM: DOS XMS Driver, Version 2.77 - 02/27/91
XMS Specification Version 2.0
COPYRIGHT 1988-1991 Microsoft Corp

Installed A20 handler number 1
64K High Memory Area is available.

Bad or missing C:\DOS\ASNI.SYS
Error in CONFIG.SYS line 5
```

As you can see, the statements that set files and buffers generate no messages. However, the statement that loads HIMEM.SYS produces a three-line message detailing the version of the driver. The next two lines confirm that you've successfully loaded DOS in the high part of memory. But the last message is most helpful: It tells you that DOS couldn't find or load the misspelled ASNI.SYS driver. Better still, it tells you that the error occurs in line 5 of the CONFIG.SYS file. With this information, you'd know to edit your CONFIG.SYS file to change *asni.sys* to *ansi.sys*. Then, when you rebooted, DOS would successfully load the ANSI.SYS driver.

Note

Once you've finished troubleshooting your system, you may find that the need to press a key to continue booting has become tiresome. When it bores you, simply edit your AUTOEXEC.BAT file to remove the now superfluous PAUSE command. ■

Continued from page 1

How your computer pulls itself up...

memory. As you might guess, the display memory helps your monitor show information that you type at the keyboard or that DOS generates. The most important part of this information is the type of display you have, usually VGA, CGA, or EGA. For example, the first message we see when our computer boots is:

```
SpeedSTAR VGA Vers. 3.00 (c) Diamond Computer
Systems, Inc
512Kb display memory installed
```

BIOS (basic input/output system)

Another bit of information your computer displays is the manufacturer and version of its BIOS chip. BIOS stands for basic input/output system and, as the name suggests, it's the lifeblood of your computer. Usually, BIOS is in read-only memory (ROM). BIOS provides instructions for your computer on a level more basic than DOS. However, DOS isn't the only program that needs BIOS; many applications communicate with BIOS, as well. In fact, we suggest that you make a note of your BIOS version before calling a technical support number for any application you've installed. Occasionally, very new versions of an application may have a problem communicating with a particular version of BIOS. If that's the case, the technical support folks will ask you for the type of BIOS you're using so they can help fix the problem. Phoenix, Award, AMI, and DTK manufacture much of the BIOS used in PCs. Here's a sample BIOS version message from our PC:

```
Phoenix 80386 ROM BIOS Version 1.10.10
Copyright (C) 1985-1989 Phoenix Technologies Ltd.
All Rights Reserved
```

The CPU (central processing unit)

If you purchased the PC yourself, you probably already know whether it uses an 8088 (also called an XT), 286 (or AT), 386, or 486. These numbers refer to the central processing unit—the microprocessor that is the workhorse of your system. However, if you just inherited the computer from someone else, you can confirm at boot up what kind of microprocessor it uses. Generally, higher-numbered chips are able to work more quickly. (The chip's speed, which is rated in megahertz (MHz), is another factor in how fast your computer works, but this information usually doesn't appear at boot up.) Depending on the application you're using, you might not notice the speed difference.

Soon after booting, your monitor will display the type of central processing unit and the company that manufactured or licensed it. For example, your processor might be an INTEL Corp 486 or MYLEX Corp 386. Here's a sample line from our PC:

```
MYLEX Corporation
386
```

Some applications, such as computer-aided design programs, need to perform lots of mathematical calculations. Computers that use these types of programs often contain a second type of processor called a math coprocessor. Math coprocessors are numbered one greater than the central processing unit, so a 386 CPU would be accompanied by a 387 math coprocessor. If you have a math coprocessor on your computer, your computer will display the message *math coprocessor installed*.

RAM (random access memory)

After displaying the CPU version, a typical system will pause as it tests its random access memory, or RAM. The RAM test quickly displays a value for your base memory, such as 640 Kb. All programs are able to access this base, or conventional, memory. If you have additional memory installed in your system, you'll see the computer count this "extended" memory twice before it displays the value and continues with the boot-up process. For example, our PC displays the following information about RAM:

```
640K Base Memory, 03456K Extended
```

To arrive at the total amount of RAM in this computer, you'd add 640 to 3456, then divide by 1024. (1024 equals one megabyte.) With this calculation, you can see that our PC has four megabytes, or 4 Mb, of RAM.

Unlike ROM, the read-only memory that stores the most basic instructions for your system, your computer can both read from and write to RAM. (Incidentally, RAM is what people usually mean when they refer to "memory." However, some people imprecisely use "memory" to describe the storage available on hard disks.) Your computer uses RAM to hold the instructions that make up the programs you're running and also to hold the documents, databases, or spreadsheets you're working with. Consequently, if your PC has too little RAM, you may not be able to run very large programs or to work with extremely large spreadsheets or databases. Also, because programs can retrieve information placed in RAM very quickly, you'll often notice that the same applications run faster on machines with more RAM. However, before you add RAM to your PC, make sure that the applications you use are able to use extended memory or that you will be able to add a memory manager program to help them take advantage of additional memory.

After completing the RAM check, you'll probably hear a hum as your system checks its diskette drive(s) and hard disk. At this point, your system is looking for two hidden files called IO.SYS and MSDOS.SYS, which DOS places on a disk when you format it with the /s switch. (Generically, these files may instead be called

IMDC:1144914 I:1406920 Q1 921
FRANK RALLY
1796 GRACE AVE
SAN JOSE CA 95125-5620

Please include account number from label with any correspondence.

IBMBIO.COM and IBMDOS.COM.) Your system needs to find and load these files before it can carry out the commands in the CONFIG.SYS file.

Configuring the system

Once your computer has finished checking its chips, it goes on to get instructions from the CONFIG.SYS file. Although it contains a series of instructions for DOS, the CONFIG.SYS isn't a batch file. Usually, a batch file issues the same commands that you could issue at the DOS prompt, but most commands used in the CONFIG.SYS file can't be issued from the prompt. These special configuration commands tell DOS how to manage your system's memory, keyboard, display, and other devices attached to your system.

On many PCs, the CONFIG.SYS file tells DOS how to use part of RAM. For example, an application might need to reserve extra storage space in RAM. The program would use these spaces, called *buffers*, as a holding area when it reads from and writes to the disk. A database program, for example, might require you to reserve 30 buffers by placing the following line in your CONFIG.SYS file:

```
buffers=30
```

The CONFIG.SYS file also installs *device drivers*, which contain special instructions that allow DOS to communicate with devices attached to your system. For example, if you attach a CD-ROM drive to your system, you'll need to tell DOS where to look for the file that drives the CD-ROM.

Like the CONFIG.SYS file itself, these special drivers end with the SYS extension. Sometimes, the "devices" they drive are actually within the computer, so you can't see them. For example, if your computer contains more than one megabyte of RAM, you might tell DOS to look for the HIMEM.SYS driver. HIMEM.SYS, available in DOS versions 4.0 and 5.0, will help manage your extended memory. If you use HIMEM.SYS, you also can create a cache with SMARTDRV.SYS. Consequently, many computers with extended memory have the following lines in their CONFIG.SYS files:

```
device=c:\dos\himem.sys
device=c:\dos\smartdrv.sys
```

Obviously, we've just given you a small sampling of what the CONFIG.SYS file does. We'll cover specific topics, such as creating a disk cache, in future issues of *Inside DOS*. (You might want to look at Van Wolverton's article "Modifying CONFIG.SYS to Take Advantage of DOS 5's Memory Enhancements," which appeared in the September 1991 issue of *Inside DOS*. Or, if you're interested in an introduction to this file, see Van's article "An Overview of CONFIG.SYS" in the September 1990 issue.)

After completing the instructions in the CONFIG.SYS file, DOS runs COMMAND.COM. This command, which is in your root directory, is called the "command interpreter." In other words, it's the part of DOS that lets you use all of the other commands, such as DIR and PROMPT.

The AUTOEXEC.BAT file

In the third phase of booting, your computer executes any commands you've placed in your AUTOEXEC.BAT file. As with any batch file, you can place any command in AUTOEXEC.BAT that you can run from DOS. Usually, AUTOEXEC.BAT files contain commands to set your prompt and path. For example, the lines

```
prompt $p$g
set path=c:\dos;c:\batch;c:\mouse;c:\word
```

set the prompt as the current directory followed by the greater-than sign and tell DOS to look for commands in the C:\DOS, C:\BATCH, C:\MOUSE, and C:\WORD directories. If you use these commands in your AUTOEXEC.BAT file, DOS will display the current drive and path in the prompt and will look for commands in the \DOS, \BATCH, \MOUSE, and \WORD directories on your C: drive. Van Wolverton gives more details about AUTOEXEC.BAT in "Take Advantage of AUTOEXEC.BAT to Customize Your System," which appeared in the November 1991 issue of *Inside DOS*.

Conclusion

In this article, we've looked at the steps your computer goes through when you boot it. Understanding these basic steps can help you troubleshoot your computer. We've also defined some common terms that describe your computer's hardware, such as RAM and BIOS. Knowing these terms can help you compare computers when you're shopping for one. ■

Printed on Halopaque: 100% acid free, made from 50% de-inked, recycled fiber including 10% post-consumer waste. This journal is 100% recyclable.